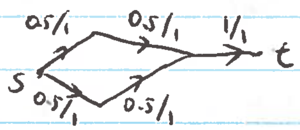
Recall:

F-F algorithm finds the max flow in $O(m^2 \cdot k)$
where k is the largest capacity of an edge.
F-F can be used to find min-cut.

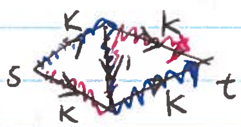
Question 1: (Recall that all Capacities are integer)
Is it possible to have max flow that assign no-integer values to some edges?
$f: E \to \mathbb{R}^+$?
Yes:



Question 2: Is there always an all integer max-flow?
Yes. b/c F-F always output integer valued flow.
and we know it finds max-flow.

Remark: The running time is not efficient when k is large number.
Input size: $\Theta(m \underbrace{\log k}_{\text{number of bit to write a number between 1 to k}})$ bits

(This is an exponential algorithm)



A Faster FF:

Possible approaches

① Always pick the shortest path from s to t
Leads to an efficient alg.

② Fattest P (Find the path with Largest bottleneck)

We will use a variation of ①, ②

High-Level description:

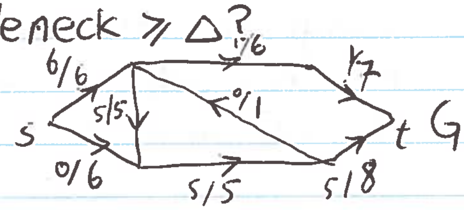
- Initialy set $\Delta = 2^{\lceil \log_2 K \rceil}$, that is Δ is the smallest power of 2 that is at least K.
  (eg if $K=13$ $\Delta=16$, if $K=17 \Rightarrow \Delta=32$)

While there are augmenting path with bottneck $\geq \Delta$. Use them to augment the flow.

- When we run out of these. set $\Delta \leftarrow \Delta/2$ and go back to while loop.
- If $\Delta=1$, then stop.

How can we find out if there are any path with bottleneck $\geq \Delta$?

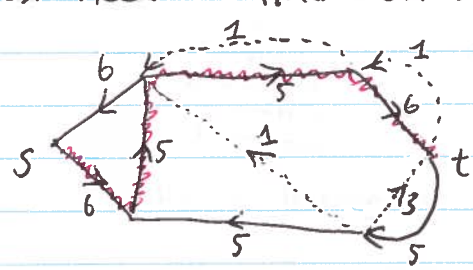
Ex:



$\Delta=4$

Let $G_f(\Delta)$ be the subgraph of G_f consisting only of the edges with residual cap $\geq \Delta$.

We just need to find an s-t path in $G_f(\Delta)$.



$G_f(\Delta)$

Scaling F-F-algorithm

set $\Delta = 2^{\lceil \log_2 K \rceil}$ where K is the larges capacity

set $f \equiv 0$, construct G_f

```
while Δ≥1
    while ∃ an s-t path P in Gf(Δ)
        Augment(f, p)
        Update Gf
    endwhile
    Δ ← Δ/2
end while
```

Running Time:

Find s-t path P in $G_f(\Delta)$ $= O(m)$

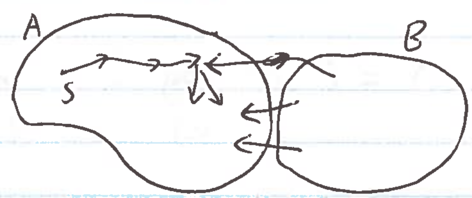
We need to understand the number of iterations.

- The outer loop has $\lceil \log_2 k \rceil$ iterations.
- The inner loop? How many iterations in Δ-phase?

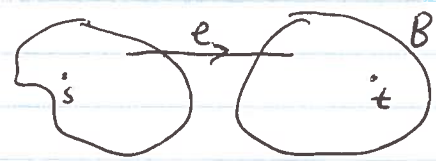
Claim: Let f be the flow at the end of Δ-phase (when no s-t paths in $G_f(\Delta)$)

There is a cut (A, B) s.t. maxflow $\leq Cap(A, B) \leq val(f) + m\Delta$

Pf: Let A be all node reachable from S in $G_f(\Delta)$

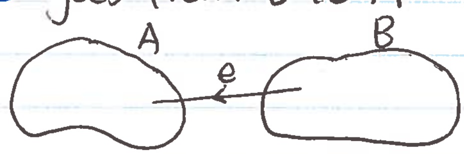


If e is an edge from A to B in the original network:



$f(e) > C_e - \Delta$ $\qquad$ $C_e - f(e) < \Delta$

If e goes from B to A



$f(e) < \Delta$

$$Val(f) = f^{out}(A) - f^{in}(A) = \sum_{\substack{e \text{ from} \\ A \text{ to } B}} f(e) - \sum_{\substack{e \text{ from} \\ B \text{ to } A}} f(e)$$

$$\geq \sum_{\substack{e \text{ from} \\ A \text{ to } B}} (C_e - \Delta) - \sum_{\substack{e \text{ from} \\ B \text{ to } A}} \Delta$$

$$= \sum_{e \text{ from } A \text{ to } B} C_e - \sum_{\substack{e \text{ from} \\ A \text{ to } B \text{ or} \\ B \text{ to } A}} \Delta$$

$$Cap(A, B) - m\Delta$$

We showed:

$val(f) \geq Cap(A,B) - m\Delta \geq$ max-flow $- m\Delta$

Let's look at the flow at the end of the previous phase

$Val(f_{prev}) \geq$ max-flow $- 2\Delta m$

How many augmentations can we have in Δ-phase?

We can have at most $2m$ augmentations.

b/c each one increases the value by at most Δ and starting from max-flow $- 2m\Delta$, we cannot go above max-flow.

Total running time

$O((\log_2 k) \times m \times m) = O(m^2 \log_2 k)$ instead of

$O(m^2 k)$ with the naive F.F.

Remark: This is a special instance of FF $\Rightarrow$ it finds max-Flow